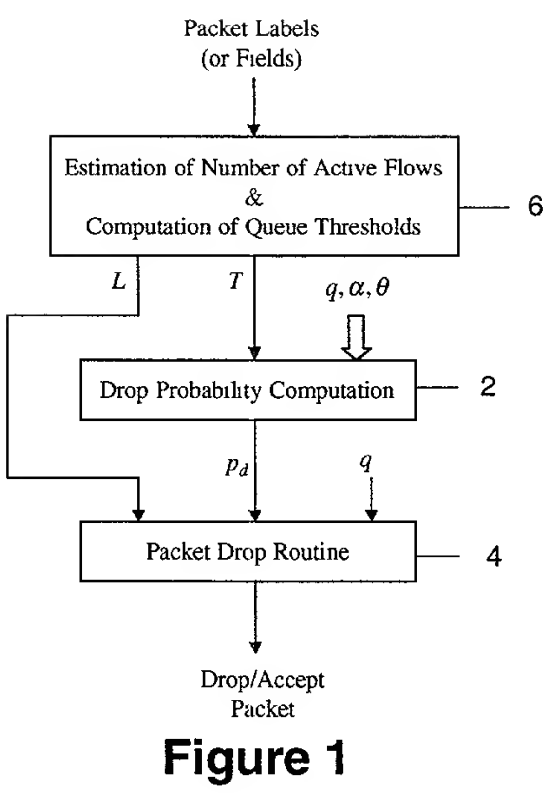
Packet Labels
(or Fields)
Estimation of Number of Active Flows
&
Computation of Queue Thresholds
6
L
T
q, α, θ
Drop Probability Computation
2
p_d
q
Packet Drop Routine
4
Drop/Accept
Packet

Figure 1

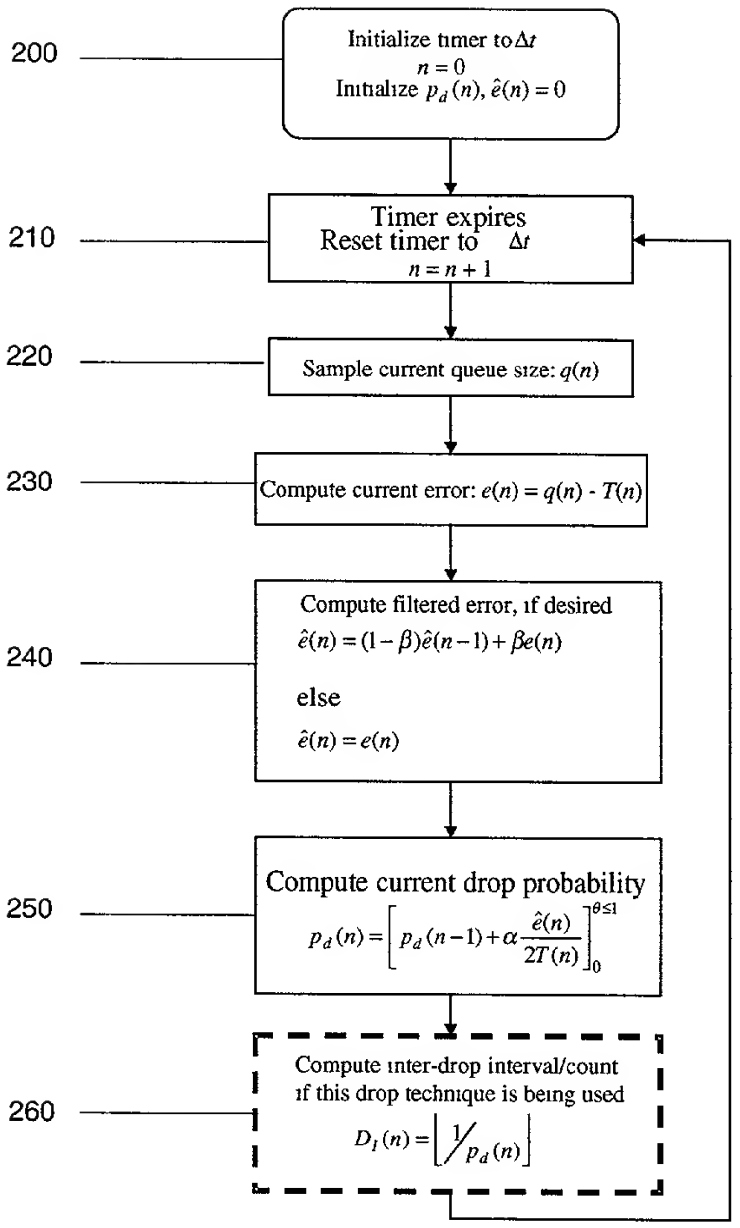
200
Initialize timer to Δt
$n = 0$
Initialize $p_d(n), \hat{e}(n) = 0$
210
Timer expires
Reset timer to Δt
$n = n + 1$
220
Sample current queue size: $q(n)$
230
Compute current error: $e(n) = q(n) - T(n)$
240
Compute filtered error, if desired
$\hat{e}(n) = (1-\beta)\hat{e}(n-1) + \beta e(n)$
else
$\hat{e}(n) = e(n)$
250
Compute current drop probability
$p_d(n) = \left[p_d(n-1) + \alpha \frac{\hat{e}(n)}{2T(n)} \right]_0^{\theta \le 1}$
260
Compute inter-drop interval/count
if this drop technique is being used
$D_I(n) = \left\lfloor 1/p_d(n) \right\rfloor$

Figure 2

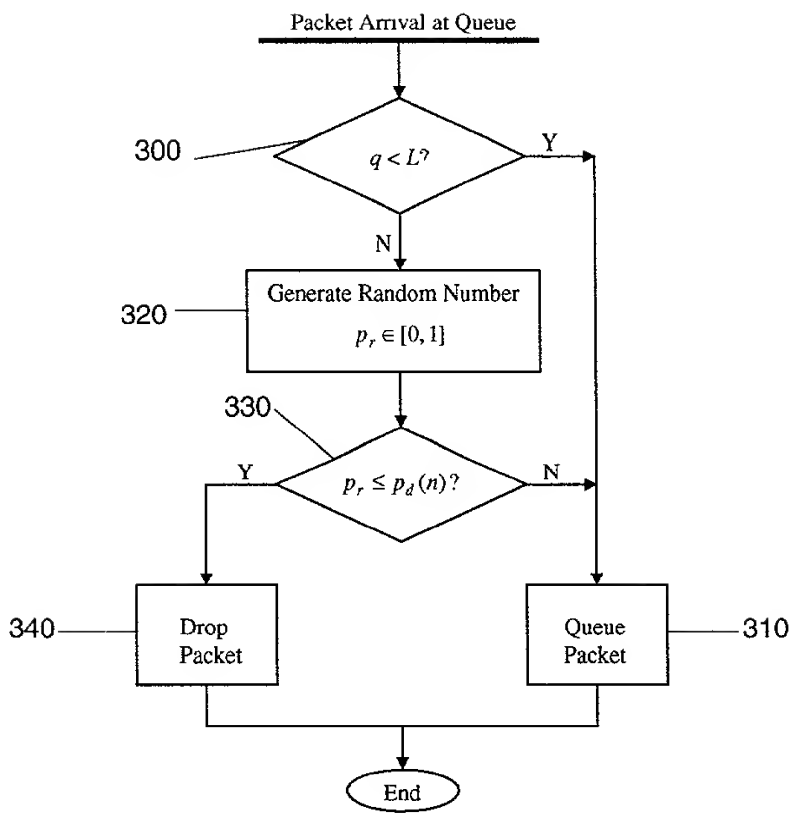
Packet Arrival at Queue
300
$q < L$?
Y
N
320
Generate Random Number
$p_r \in [0, 1]$
330
Y
$p_r \leq p_d(n)$?
N
340
Drop Packet
Queue Packet
310
End

Figure 3

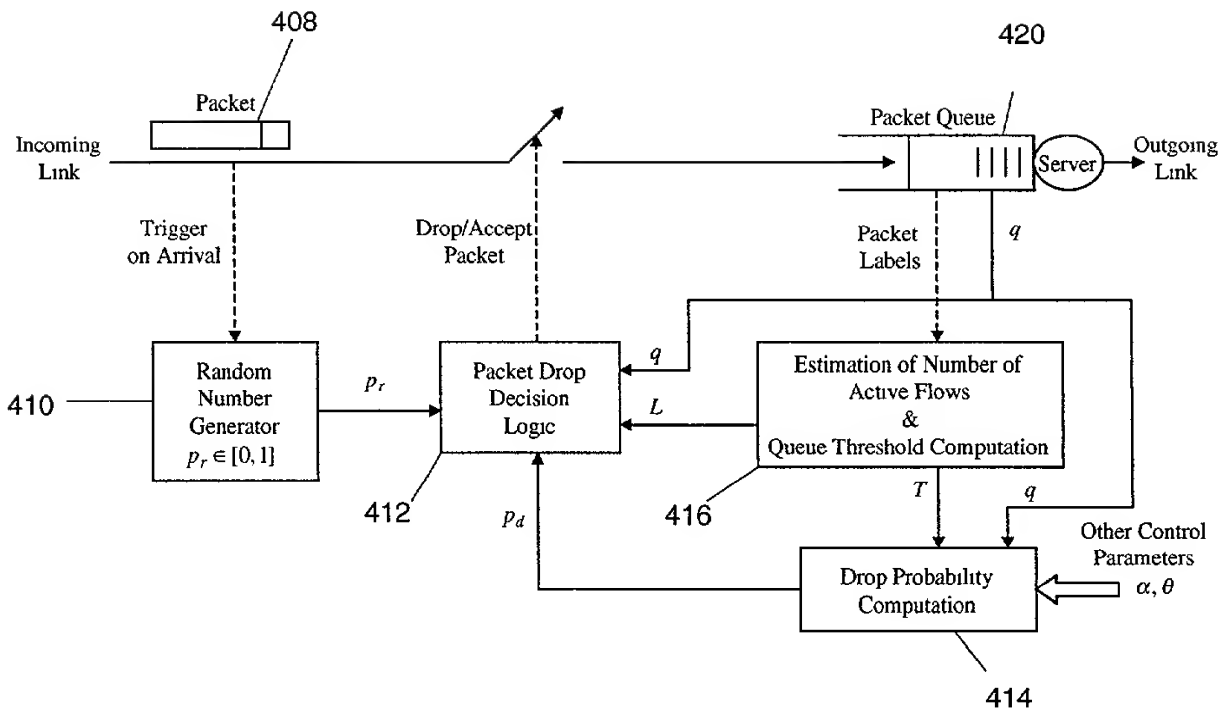
408
420
Packet
Incoming Link
Packet Queue
Server
Outgoing Link
Trigger on Arrival
Drop/Accept Packet
Packet Labels
q
410
Random Number Generator
$p_r \in [0, 1]$
p_r
Packet Drop Decision Logic
q
L
Estimation of Number of Active Flows & Queue Threshold Computation
412
p_d
416
T
q
Other Control Parameters
α, θ
Drop Probability Computation
414

Figure 4

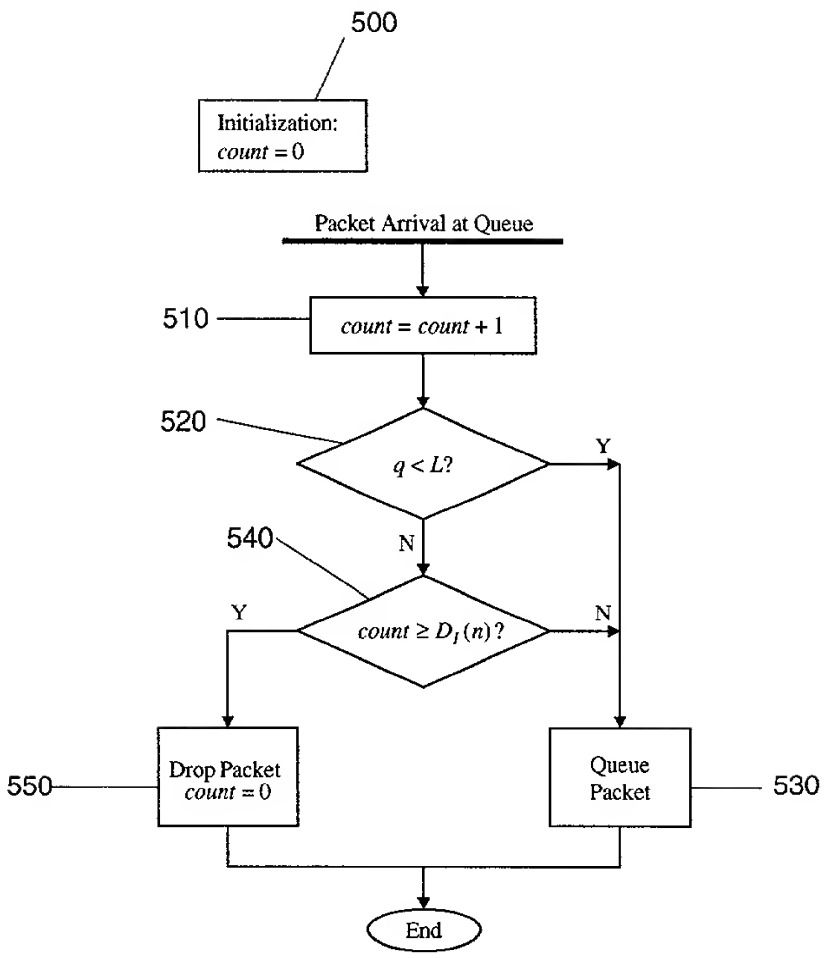
500
Initialization:
count = 0
Packet Arrival at Queue
510
count = count + 1
520
q < L?
Y
N
540
Y
count ≥ D_I(n)?
N
550
Drop Packet
count = 0
Queue
Packet
530
End

Figure 5

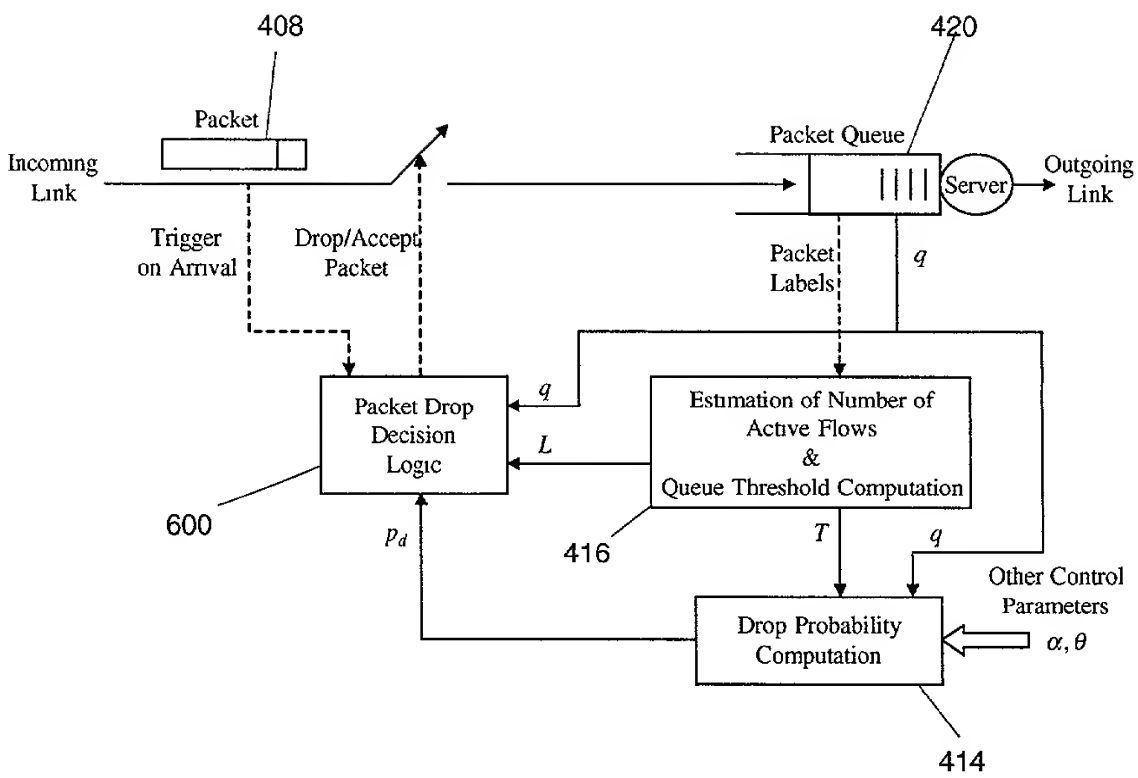
408
420
Packet
Incoming Link
Packet Queue
Server
Outgoing Link
Trigger on Arrival
Drop/Accept Packet
Packet Labels
q
Packet Drop Decision Logic
Estimation of Number of Active Flows & Queue Threshold Computation
L
600
p_d
416
T
Other Control Parameters
Drop Probability Computation
α, θ
414

Figure 6

Define variables:

- bit_vector /* bit vector indexed by hash functions */
- session_ID /* e.g., source/destination addresses and port numbers */
- count /* to count number of new sessions */
- measurement_interval /* time interval over which sessions are counted */
- $hash_i$_index /* hash index for hash function $hash_i$, i, (i = 1, 2,..... N) */

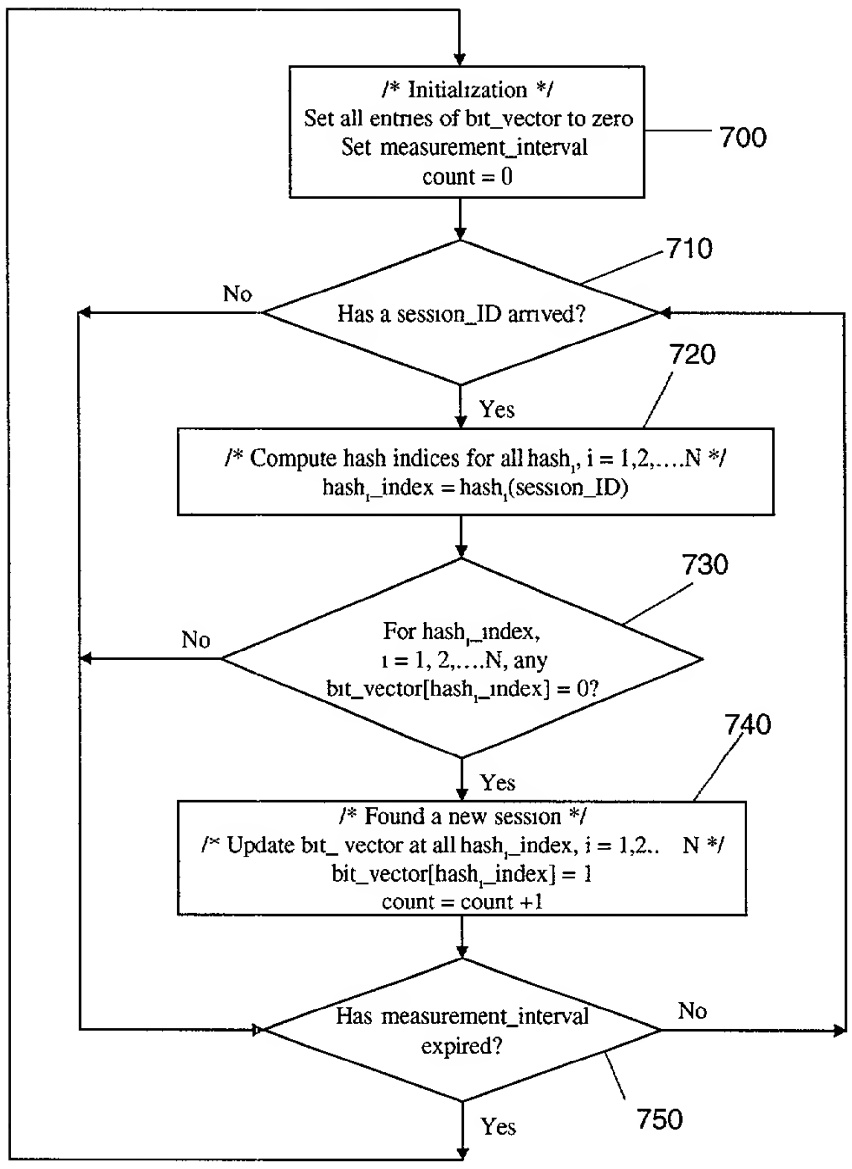

Figure 7

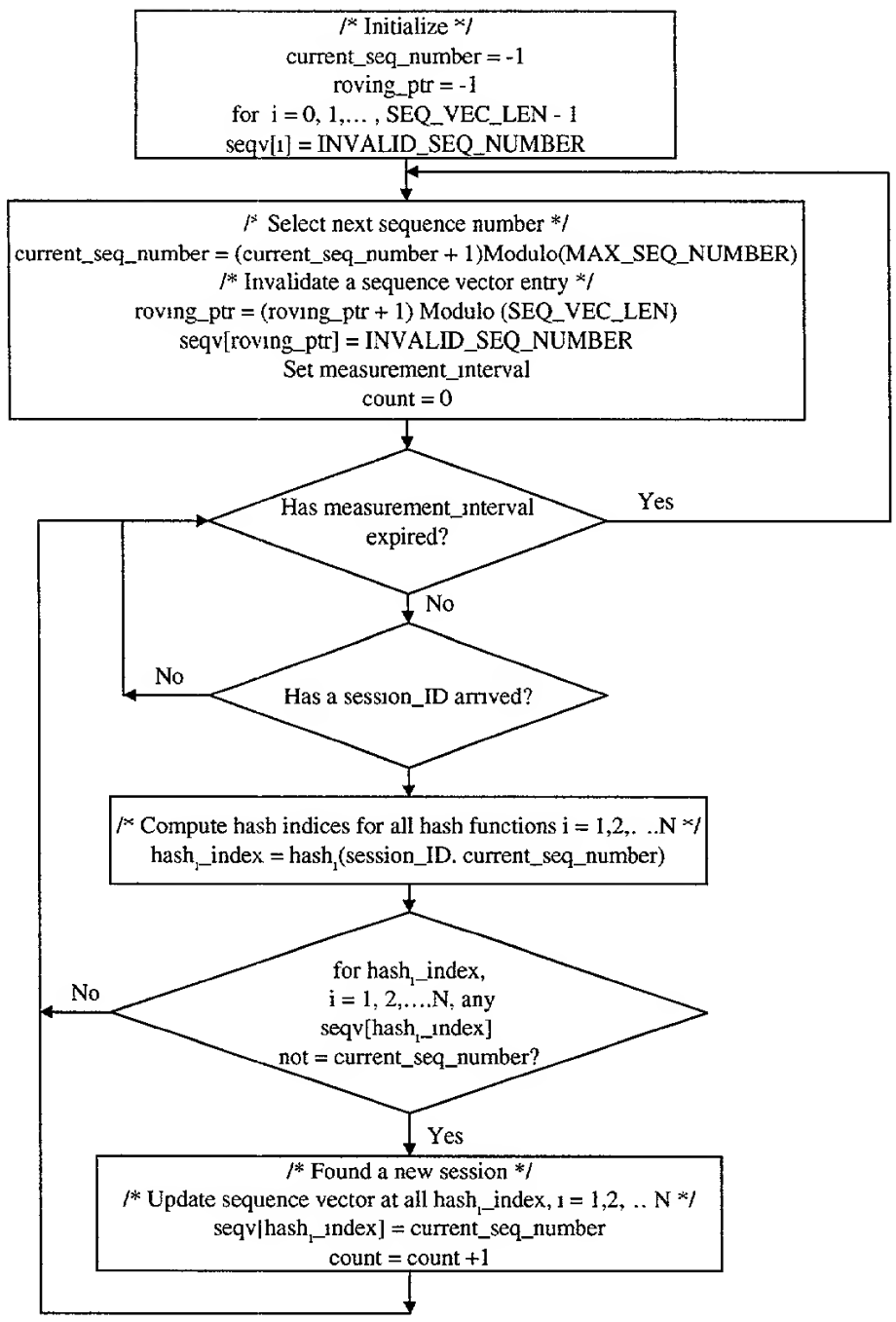
/* Initialize */
current_seq_number = -1
roving_ptr = -1
for i = 0, 1,... , SEQ_VEC_LEN - 1
seqv[i] = INVALID_SEQ_NUMBER
/* Select next sequence number */
current_seq_number = (current_seq_number + 1)Modulo(MAX_SEQ_NUMBER)
/* Invalidate a sequence vector entry */
roving_ptr = (roving_ptr + 1) Modulo (SEQ_VEC_LEN)
seqv[roving_ptr] = INVALID_SEQ_NUMBER
Set measurement_interval
count = 0
Has measurement_interval expired?
Yes
No
No
Has a session_ID arrived?
/* Compute hash indices for all hash functions i = 1,2,. ..N */
hash_i_index = hash_i(session_ID. current_seq_number)
No
for hash_i_index,
i = 1, 2,....N, any
seqv[hash_i_index]
not = current_seq_number?
Yes
/* Found a new session */
/* Update sequence vector at all hash_i_index, i = 1,2, .. N */
seqv[hash_i_index] = current_seq_number
count = count +1

Figure 8

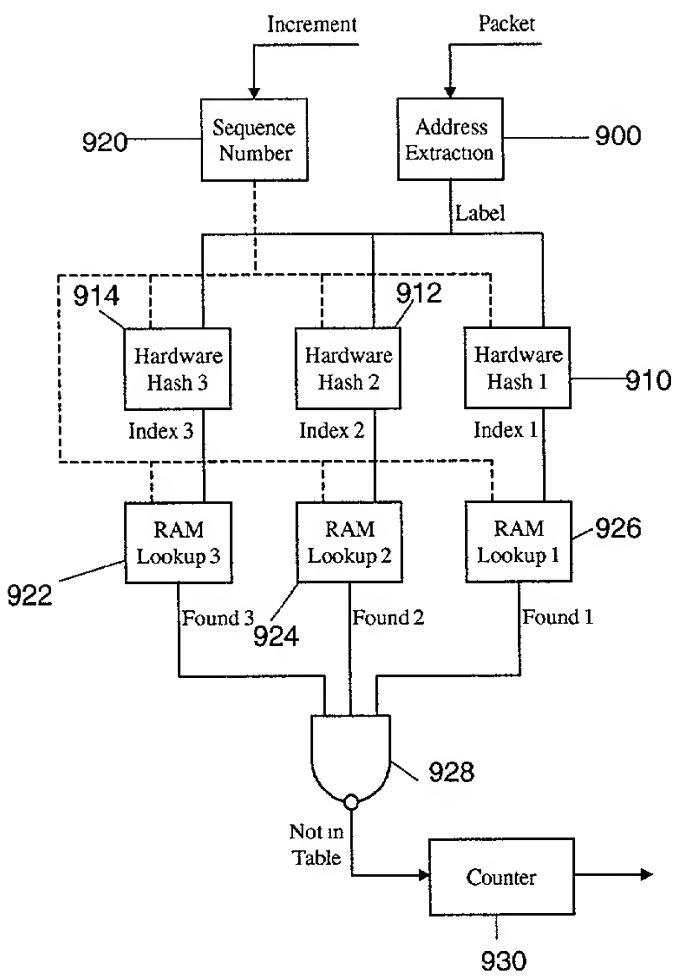
Increment
Packet
Sequence Number
920
Address Extraction
900
Label
914
912
Hardware Hash 3
Hardware Hash 2
Hardware Hash 1
910
Index 3
Index 2
Index 1
RAM Lookup 3
RAM Lookup 2
RAM Lookup 1
926
922
Found 3
924
Found 2
Found 1
928
Not in Table
Counter
930

Figure 9